

Parakeet Virtual Cable Concept Demonstrator

A.B. Reynolds and W.D. Blair

DSTO-TR-1113

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

20010718 101

Parakeet Virtual Cable Concept Demonstrator

A.B. Reynolds and W.D. Blair

**Communications Division
Electronics and Surveillance Research Laboratory**

DSTO-TR-1113

ABSTRACT

Project Parakeet is deploying a secure military telephony system into Australia's land forces. The Battlespace Command Support System (BCSS) project is deploying local area networks (LAN) and computers to these same forces. This report describes a concept demonstration that permits the use of the BCSS LAN for the connection of Parakeet telephones to the Parakeet circuit switches rather than using separate cabling. Such a device would provide for faster setup/teardown of deployed headquarters and reduce the demands on linesmen to lay cable within the headquarters.

RELEASE LIMITATION

Approved for public release

DEPARTMENT OF DEFENCE
DEFENCE SCIENCE & TECHNOLOGY ORGANISATION **DSTO**

AQ FOI-10-1778

Published by

*DSTO Electronics and Surveillance Research Laboratory
PO Box 1500
Salisbury South Australia 5108 Australia*

*Telephone: (08) 8259 5555
Fax: (08) 8259 6567
© Commonwealth of Australia 1999
AR-011-778
March 2001*

APPROVED FOR PUBLIC RELEASE

Parakeet Virtual Cable Concept Demonstrator

Executive Summary

When an army is in the field, it needs to be able to communicate with its units and other forces. To support these communications, the Australian land forces are fielding a new trunk communications system through Project Parakeet. A key service provided by Parakeet is secure telephony. In the meantime, the Army is also fielding an element of computer automation called the Battlefield Command Support System (BCSS). It provides notebook PCs and a local area network (LAN) to connect them together in the local area.

There is a problem with these two disparate systems within the headquarters. Both require communications assets to be deployed around headquarters, and they use two separate systems to do this (field telephone cable or fibre optic for Parakeet and independent LAN cabling for BCSS). This dual cabling requires more materials and deployment time than an integrated system.

The idea of reducing this overhead, and to bring forward an emerging concept of a multi-service Local Area Subsystem, has led to the idea of the Parakeet Virtual Cable (PVC). Its purpose is to tunnel the Parakeet telephone's connection over the BCSS LAN. By having two devices, one between the telephone and the BCSS LAN, and the other between the BCSS LAN and the Parakeet multiplexer/switch the existing BCSS LAN infrastructure can be used to transport voice calls.

The key challenge addressed in this effort is caused by the difficulty in supporting a constant bit rate, real-time service over a packet based LAN which has been designed to provide for non-real-time quality of service to the data terminals. The report describes a microprocessor based solution with an interesting dual stage buffering approach to meet the demanding requirements of the telephony service on the LAN.

The PVC offers a way to use the existing ethernet infrastructure deployed by BCSS for transporting Parakeet voice calls within a headquarters. This will decrease the deployment times for cabling a headquarters environment and it will enable a transition from the circuit switched world of Parakeet to the packet switched world of data transport. The design discussed in this report has been implemented in a concept demonstrator and has been shown to work well under normal operating conditions. Opportunities exist for contractors to continue this work and then to provide the Australian Defence Force with a fielded capability.

Authors

A.B. Reynolds

Communications Division

Alfred Reynolds is a Research Engineer in Network Architecture Group of the Defence Science and Technology Organisation's (DSTO) Communications Division having joined DSTO after completing a Systems Engineering Degree at the Australian National University. He is involved in research into tactical communication systems.

W.D. Blair

Communications Division

Bill Blair is a Senior Research Scientist in Network Architecture Group of the Defence Science and Technology Organisation's (DSTO) Communications Division having joined DSTO from the Australian Army. He is involved in research into networking techniques to control Quality of Service (especially Asynchronous Transfer Mode - ATM) for military strategic and tactical communications.

Contents

1. BACKGROUND	1
2. CONCEPT	1
3. REQUIREMENTS	2
4. APPROACH	2
4.1 CDP to 4 Wire TTL Synchronous	2
4.2 4 Wire TLL to Processor	3
4.3 Processor to Ethernet	3
5. HIGH LEVEL DESIGN	4
5.1 RS232/Synchronous Speed Differential	4
5.2 Packet and Buffer Design	5
6. FUTURE DESIGN WORK	6
7. CONCLUSIONS	6
8. REFERENCES	6
APPENDIX A: OTHER HARDWARE OPTIONS CONSIDERED	7
1. 4 WIRE TLL TO PROCESSOR	7
2. PROCESSOR TO ETHERNET	7
APPENDIX B: BUFFER CONCEPTS	9
1. PIC AND UCSIMM BUFFER CONCEPTS	9
1.1 Cadence Concept	10
1.2 Reset PIN	10
1.3 Read Pointer Bounce Back	10
2. EXAMPLES	11
2.1 Normal Operation	11
2.2 Packets Dropped by the Network	12
2.3 Delayed Packets	13
2.4 Extremely Late Packets	14
APPENDIX C: MORE DETAILED INFORMATION	15
1. DETAILED DESIGN	15
1.1 Mitel Chip interface	15
1.2 16kb/s sync TTL stream to RS232	15

1.2.1	TTL Input to TX on RS232	16
1.2.2	RX on RS232 to TTL Output	16
1.2.3	Frame Pulse	16
1.3	PIC Chip program.....	17
1.3.1	Operation.....	17
B.1.3.1.1	Buffering	17
B.1.3.1.2	Interrupt Service Routine	17
B.1.3.1.3	Main Loop	18
1.3.2	Extra Features.....	18
1.4	uCsim Design	18
1.4.1	The uCsim	18
1.4.2	Kernel.....	20

2.DIFFERENCES BETWEEN SWITCH/MULTIPLEXER END AND DVT(DA) END20

3. DEVICE COST BREAKDOWN 21

4. ISSUES FOR INCLUSION IN OPERATING PROCEDURES..... 22

5. FUTURE DESIGN WORK..... 24

1. SWITCH SERVER NEGOTIATION..... 24

2. DEVICE CONFIGURATION..... 24

3. CABLING 24

4. LOOP GROUP PVC 25

1. Background

When an army is in the field, it needs to be able to communicate with its units and other forces. There are currently two types of communication systems used for this purpose, the combat net radio and the trunk communications system. For the Australian Army, the trunk communications system is Parakeet. Parakeet is a military standard circuit switched system similar in concept to civilian Narrow-band Integrated Services Digital Network. Parakeet offers 16kb/s (kilobits per second) channels to the end user, which can be used for either voice (using a military standard Continuously Variable Slope Delta Modulation, CVSD, coding scheme) or data.

As technology advances it is possible to automate business processes, and for the Army this involves deploying PCs with command and control software. For the Australian Army this system is called the Battlefield Command Support System (BCSS). It provides notebook PCs and an ethernet local area network (LAN) to connect them together in the local area. (Wide area connections are via routers that run over the Parakeet or via gateway PCs connected to combat net radio.)

There is a problem with these two disparate systems within the headquarters. Both require communications assets to be deployed around headquarters, and they use two separate systems to do this (field telephone cable or fibre optic for Parakeet and independent LAN cabling for BCSS). This dual cabling requires more materials and man-hours than an integrated system.

2. Concept

The current parakeet trunk network uses a Digital Voice Terminal/Data Adaptor, DVT(DA), as the telephone terminal for end users. Every DVT(DA) needs to have a 2 wire cable (known as WD-1/TT) laid to it from the Parakeet multiplexer or switch assembly to enable communications. The act of laying cable costs man-hours and slows the headquarters deployment. At the same time as deploying this cabling, an ethernet local area network (LAN) for the BCSS is also being laid. The process of laying two separate yet similar networks has led to the idea of the Parakeet Virtual Cable (PVC). Its purpose is to tunnel the DVT(DA)'s connection over the ethernet network. By having two devices, one between the DVT(DA) and the ethernet, and the other between the ethernet and the multiplexer/switch, the ethernet network infrastructure can be used to transport voice calls in addition to data traffic.

Ethernet is not a reliable medium and packets can be lost or delayed in the network whereas such loss or delay does not occur on normal telephone cable. The PVC must also be able to transport the constant bit rate (CBR) stream emerging from the DVT(DA) to the switch or multiplexer over the BCSS ethernet, and it must be able to do this with a high level of reliability.

3. Requirements

The main requirement is the use of the ethernet infrastructure as the transport for the DVT(DA) bit stream. There are some constraints on the performance of the network and the interface device's speed and size.

The device must be able to provide two wire connections to the Parakeet elements (DVT(DA) at one end and circuit switch/multiplexer at the other). It must convert the bit stream into ethernet packets that will be transported over the LAN. The system must also be able to handle brief interruptions in network availability (because of traffic or physical problems). It should be able to work when exposed to normal ethernet latency (ie end to end transmission delay) and jitter (delay variation), it should also be able to negate their effects.

The network must be able to tunnel the 16kb/s Parakeet streams when the BCSS network is under normal operating conditions.

The device must also be able to be deployed in the BCSS environment and it should require a minimum of setup.

There are two distinct problems to be solved in the design of the PVC:

- Connection of the Parakeet elements to a microprocessor. In itself, this requires:
 - conversion of the two wire signal (Conditioned Di-phase [CDP]) used by Parakeet into a four wire¹ synchronous bitstream of the correct voltage levels (TTL) and
 - getting the bitstream into a processor (typically wanting to manipulate bytes rather than bits).
- The packetisation and transmission of the data over an ethernet LAN with processing the handle latency, jitter and packet loss.

The various concepts considered and the path taken for each problem will now be discussed. Except for minor points discussed in the detailed design, the two end devices (ie switch/multiplexer to ethernet and DVT(DA) to ethernet) are identical. For the purposes of clarity of the descriptions, this report will refer to the DVT(DA) to ethernet end.

4. Approach

4.1 CDP to 4 Wire TTL Synchronous

The possible solutions to this problem are limited by the method that the DVT(DA) produces its two wire signal. The DVT(DA) uses a chip created by Mitel Semiconductors² that converts a four wire TTL signal (used internally by the digital

¹ Four wire in this context means one receive signal and its clock, and one transmit signal and its clock.

² Mitel Semiconductors are an integrated circuit manufacturer for communications products. More information can be found on their website at <http://www.mitelsemi.com/>.

side of the analog to digital conversion) into a two wire CDP signal with special characteristics. The only viable way to convert this signal back into the original four wire signal is to use another Mitel IC that is designed to demodulate this signal. Therefore, this section of the device was designed around using this IC.

4.2 4 Wire TTL to Processor

Appendix A describes a few options that were explored for this problem. The speed of the TTL synchronous data stream is 16kb/s. This is relatively slow in terms of electronic circuits, so it will not require sophisticated hardware to read this into the processor.

The approach taken is to convert the four wire TTL stream into a RS232 asynchronous byte oriented data stream. By converting the signal into this common format, it is possible to interface this part of the design into a number of different ethernet interface devices. The flexibility offered by converting the data stream into an industry standard means that the synchronous to asynchronous device could also be used in other applications. The key hardware element is the PIC chip³ that is a programmable synchronous/asynchronous interface with quite sophisticated buffer management. The exact implementation of this part is discussed further in the following sections of this paper.

4.3 Processor to Ethernet

This problem also lends itself to many solutions. The possible solutions range from using TTL gates to interface to the UTP cable through to using a full desktop PC. There were three main possibilities that were considered:

- Using the BCSS laptop
- Using custom designed card employing a discrete Ethernet integrated circuit
- Using a small embedded PC

An embedded PC was determined to be the only reasonable alternative. This option still leaves many characteristics to be determined. There is the size of the device, the power requirements, the interfaces it has, the operating system it runs, the processing power, the cost per unit, the cost of licensing a software development kit (SDK) and the development time. All these factors need to be considered before a device is chosen.

The device must be portable and it must not take up much space on the user's desk, so the dimensions of the embedded device must be less than that of a typical lunchbox. The device will have access to a constant power supply, so low power operation is not required but it would be a useful attribute. It must have an ethernet interface and a serial port interface. It must also have some way to configure the device. The operating system it runs must guarantee a minimum latency which is in the range of 2 msec, and it must be able to process a full duplex 16kb/s CBR stream. The operating system must be highly reliable so that the device can run for weeks without any user intervention. It must be reasonably cheap per unit, as two devices per DVT(DA) are required. DSTO has limited resources, so the device must not have

³ A PIC chip is a programmable microprocessor made by MicroChip. More information on PIC chips can be found at <http://www.microchip.com>

large operating system or SDK costs. Finally, a minimum of development time must be spent in designing the programs to run on the device.

These requirements for the embedded PC lead to two possible solutions. Embedded PCs based on the VxWorks operating system and Linux based devices. After costing the various solutions, the uCsim⁴ was selected. It consists of a 16MHz 68EZ328 DragonBall Microcontroller, 8MB of RAM, 2MB of flash ROM, a RS232 serial port, a 10baseT ethernet port, low power consumption and 18 input/output lines. The uCsim runs the Linux operating system so there is no cost involved in getting the operating system and the source code is available if anything needs to be altered. The uCsim also has the ability to operate as a real time device if the standard Linux operating system does not meet the latency requirements of the system.

5. High Level Design

The final design for each end device consists of three major elements with total cost in the order of \$400 per end. The three elements are the two wire to four wire converter, the 4 wire to RS232 converter and the RS232 to ethernet stage. The data flows in this design as described by this diagram:

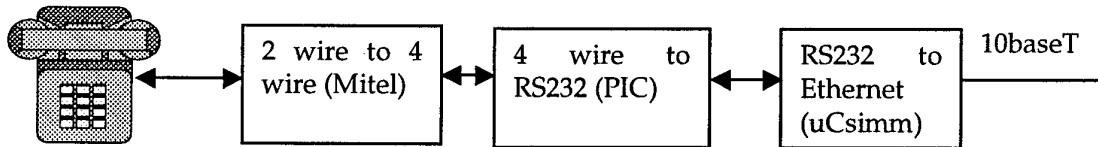


Figure 1 – High Level Design (with Key Integrated Circuit Element)

Various general issues and features with this design will now be discussed. More detail including device costings and procedural issues are given at Appendix B and C.

5.1 RS232/Synchronous Speed Differential

There is a deliberate difference in the speed of the RS232 port (56kb/s) and the four wire synchronous port (16kb/s). This speed difference is required because:

- At a minimum the asynchronous link must operate at 20kb/s to overcome the extra overhead of start and stop bits used to delineate byte boundaries on the asynchronous line.
- The asynchronous line can clear a packet of data at a higher speed than the synchronous port and this helps to overcome possible PIC buffer underflows caused by delays in the uCsim from operating system scheduling.
- The PIC and the uCsim need to have an asynchronous bit rate that can be supported by both devices. In our case, differences in clock crystal rates led to 56kb/s being the optimal rate.

⁴ More information about the uCsim is contained later in this document, and can also be found at <http://www.uclinux.org/uCsim/>

5.2 Packet and Buffer Design

More detail is provided in Appendix B. The PVC design uses a packet carrying 35msec audio data which results in a packet size of 70 bytes. The structure has included a sequence number to identify whether a packet has been lost on the ethernet.

There is effectively no buffering of packets required in the Parakeet to ethernet process. Bits from the synchronous port are packetised and sent once the packet is complete. The smart buffering is placed in the ethernet to Parakeet process after the packets have traversed the ethernet where the packet delay, jitter and loss occurs.

Ethernet is a best effort transmission system so there can be no guarantees on the time it will take to transmit a packet, or if that packet is transmitted at all. However, studies on real world ethernet networks with usage similar to the BCSS environment have shown that a maximum packet jitter of 40msec can be expected, with a large majority of packets falling within a 30 msec maximum [1].

There are two functional areas in which buffering can occur in this design, in the uCsim and on the PIC chip. The uCsim has a lot of processing power and memory, whereas the PIC is a small microprocessor with only a small amount of memory available. Nevertheless, the PIC is a real time hardware device with an accurate time reference available, whereas the uCsim only has timing via the Linux operating system and thus has limited time resolution ability. This disparity between the two devices has led to two different views of the data stream. The uCsim treats the data at a packet level, whereas the PIC is concerned with each individual byte. Therefore, two levels of buffering have been conceived. In the uCsim, a packet level buffer is used to store early packets and to handle large amounts of jitter. In the PIC, a two-packet buffer is used, with one buffer being filled by the RS232 link and one being emptied by the synchronous link to the Mitel/DVT(DA).

If a voice packet has not arrived by the time the PIC playout buffer is empty, then a silent packet must be inserted into the synchronous stream to maintain the transmission to the Mitel/DVT(DA). If the packet has been lost on the ethernet, then there has been no additional data added to the synchronous stream. However if a late packet subsequently arrives and is queued in the uCsim for later transmission to the PIC an irreducible time delay will be inserted in the stream. If the addition of silent packets along with the real packet occurs often then the system will introduce an increasing and unacceptable end to end latency. The PVC allows for only one extra packet to be inserted, with further instances resulting in packets being discarded. The logical place for the packet discard to occur is in the uCsim but the lack of a real time clock prevents the uCsim from being able to identify whether the packet has arrived within the PIC dejitter margin. A concept we called "cadence" has been developed so that the real time clock accuracy of the PIC can be signalled to the uCsim to assist in identifying packets for discard.

6. Future Design Work

Appendix D discusses outstanding design issues that would be required for a full fielding of the PVC device. In addition, the appendix discusses a possible enhancement by developing a single device to replace a Parakeet multiplexer and connect a loop group directly to an ethernet network.

7. Conclusions

The PVC offers a way to use the existing ethernet infrastructure deployed by BCSS for transporting Parakeet voice calls within a headquarters. This will decrease the deployment times for cabling a headquarters environment and it will enable a transition from the circuit switched world of Parakeet to the packet switched world of data transport.

The design discussed in this report has been implemented in a concept demonstrator and has been shown to work well under normal operating conditions. There exists opportunities for contractors to continue this work and then to provide the Australian Defence Force with a fielded capability.

8. References

- [1] Gonsalves, T.A. (1983), Packet-voice Communications on an Ethernet Local Computer Network: an Experimental Study, Computer Communication Review, Vol. 13, no. 2, pp 178-185
- [2] Michael B. Jones and John Regehr (1999) , The Problems You're Having May Not Be the Problems You Think You're Having: Results from a Latency Study of Windows NT , visited 21/8/2000, URL - <http://www.cs.virginia.edu/~jdr8d/papers/hotos7/hotos7.html>

Appendix A: Other Hardware Options Considered

1. 4 Wire TTL to Processor

The first option investigated was using the parallel port of a standard PC. The parallel port offers up to 24 digital input/output (I/O) lines that can easily handle the 16kb/s data rate required. The problem with the parallel port is the ability to reliably read in this data. The port is designed around the idea of asserting I/O lines and then waiting for a handshake signal to inform the port that the data has been read. Therefore, it is built around the idea of asynchronous exchange of data. The four wire TTL stream is synchronous in nature however, and it is not possible to guarantee that this information will always be read in by the parallel port in the time allowed.

The second option for reading in the data is to use a dedicated PC card that connects into a standard PC bus (ISA, PCI or PCMCIA). This option could guarantee that all the data would be received as interrupts on the card could be used and buffering could be performed on the card. The problems with this option are the complexity of interfacing with the PC bus, creating a driver for the card and the availability of a PC platform in the field deployment. The major application of this device would be in a BCSS environment, which means that it will have to be able to interface with a laptop PC. The BCSS laptop's (and laptops in general) only offer a PCMCIA interface, which is very costly to develop for. Also, the creation of a driver for the various possible operating systems is not trivial. Finally, the biggest problem is the uptime of the system. For this method to work the card must always be operational, which would mean that the operating system must always be running. If the PC were to crash or be turned off (possibly because of lack of power) communications via Parakeet would no longer be possible. The DVT(DA) must always be available so this option was not considered further.

2. Processor to Ethernet

It would be possible to use the RS232 and ethernet port of a BCSS laptop to create a device that could do the packetisation and delivery of the data. However there are two large problems with this method. The first is the "not always on" problem, the standard BCSS laptops are not designed to always be turned on. If the PC is off, this method would not be able to function. The second, critical problem is that the operating system of choice for the BCSS laptop (Microsoft Windows NT) could not meet the requirements of the overall system. The failure of Windows NT is in the time it could take to service the ethernet or RS232 port. Windows NT is not designed to be a real-time or near real-time operating system, so there can be no guarantees on the time it takes to service an interrupt. When the system is busy (for example, by starting up Microsoft Word), then it is common for more than 10msec [1] to pass before an interrupt is serviced. For the PVC device, a response time of 10msec is unacceptable for

this real time application. In discussions with the communications company DSpace on their software modem (which has similar requirements to the PVC device) it was revealed that to get their modem to function reliably under Windows NT they had to have a buffer of over one second worth of data. Adding a one second latency to the voice carried by the DVT(DA) is not an option.

The second option is to use a discrete ethernet chip. This could interface directly to the four wire output of the Mitel device. To configure the ethernet device on startup however you need to have external logic to input the correct register settings. This external logic would almost certainly have to be in the form of a microprocessor. If you need a microprocessor to configure the chip, it should also be used to do some smart management of the ethernet data. This solution quickly becomes a complete embedded system like the uCsim, but without the standard interfaces and pre-made operating system.

Appendix B: Buffer Concepts

1. PIC and uCsim Buffer Concepts

The PIC contains a two-packet buffer, with one side being written into from the uCsim while the other is being read out of to the Mitel/DVT(DA). This system of writing a packet down into a separate buffer removes two problems:

- The prime purpose of this buffer is to provide about one packet worth of dejitter buffer within the PIC.
- The system also assists in avoiding a clipping effect caused by the read pointer over-taking the write point and then the write pointer over-taking the read pointer. The bounce-back effect described later ensures that this cannot happen.

The design is visualised in Figure 2:

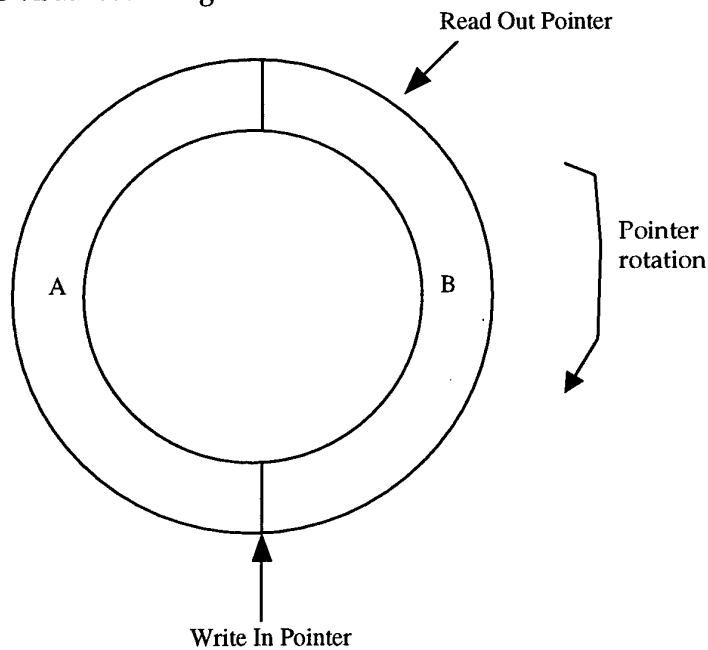


Figure 2 - Dual buffers

The role of the uCsim in this buffering is to provide another level of packet buffering, and, with assistance from the PIC, to make decisions to discard packets that are too late.

1.1 Cadence Concept

The clock available to software on the uCsim is only accurate to ± 10 msec. When the packet lengths are in this order of magnitude the internal clock cannot be used in deciding whether to drop a packet when it is too late. Since the PIC has direct access to good hardware based clock, the concept of using a heartbeat from the PIC was created. A pin on the PIC is used to indicate which half of the buffer is providing the packet currently being read out and sent to the Mitel/DVT(DA). This information can then be used by the uCsim to control the rate and timing of when packets are sent to the PIC. Each packet should be sent from the uCsim into alternate PIC buffers, ie the heartbeat must change state between each packet transmission.

Referring to Figure 2. When the read out pointer is in buffer 'A', the cadence pin on the PIC will be at the zero level. When the read out pointer is in buffer 'B', the cadence pin will be at the logical one level.

1.2 Reset PIN

The ability to reset the PIC to a known state is vital for the operation of the cadence concept. By asserting this pin on the PIC the read pointer resets to the top of the 'B' buffer and the write pointer is reset to the top of the 'A' pointer (the bottom of the circle as seen in Figure 2). On powerup the uCsim uses this reset pin to reset the PIC chip to a known state.

1.3 Read Pointer Bounce Back

As a byte of data is read out of the buffer, the value of that position is reset to a default value, in this case a "10101010" bit pattern. This pattern is chosen as it is the pattern produced out of a CVSD coder when the speaker is silent.

The read pointer must only swap buffers when a new packet has been successfully written from the uCsim. To achieve this, the read pointer has a "bounce back" idea implemented.

The write in (from the uCsim) pointer essentially moves in packet length bursts (i.e. it essentially toggles between 0 and 180 degrees on the circle) whereas the read pointer is byte based. When the read pointer reaches the end of a packet it checks whether the write pointer is at the same point. If it is then this means that there is no new data in place for the PIC to read out to the Mitel/DVT(DA) and so the read pointer is reset to a diametrically opposite position, i.e. bounced back to the start of the packet buffer just read.

The action of resetting the pointer produces two effects. The first is the insertion of a packet worth of silence into the data stream, required when the ethernet traffic has a

large jitter. The second effect is to maintain the cadence signal at the same level for two packet periods.

2. Examples

The effects of the algorithms implemented in the PIC chip and the uCsim can best be seen by the following examples. The details of the control algorithms are discussed in the detailed design section.

2.1 Normal Operation

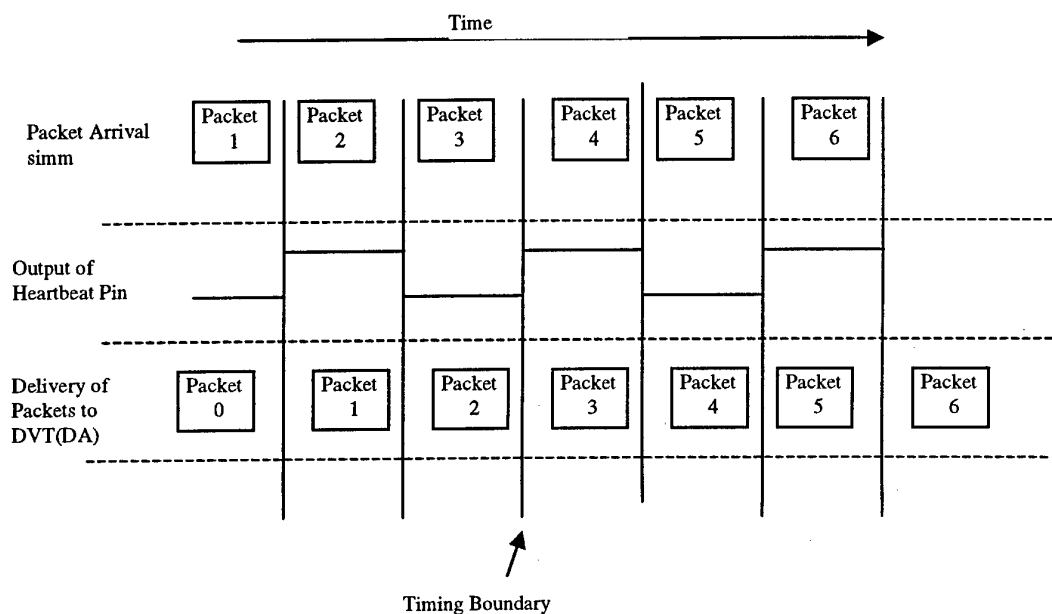


Figure 3 - Normal operation of the device

This example demonstrates the normal operation of the device. All packets arrive on time and are sent to the Mitel/DVT(DA) straight away. The cadence effect can be seen. The uCsim sees each packet arriving in turn and the PIC cadence signal is showing alternate values when the packets arrive at the uCsim indicating that they can be written into the PIC. Note that only the digital stream is passed to the PIC, there are no packet numbers available to it.

2.2 Packets Dropped by the Network

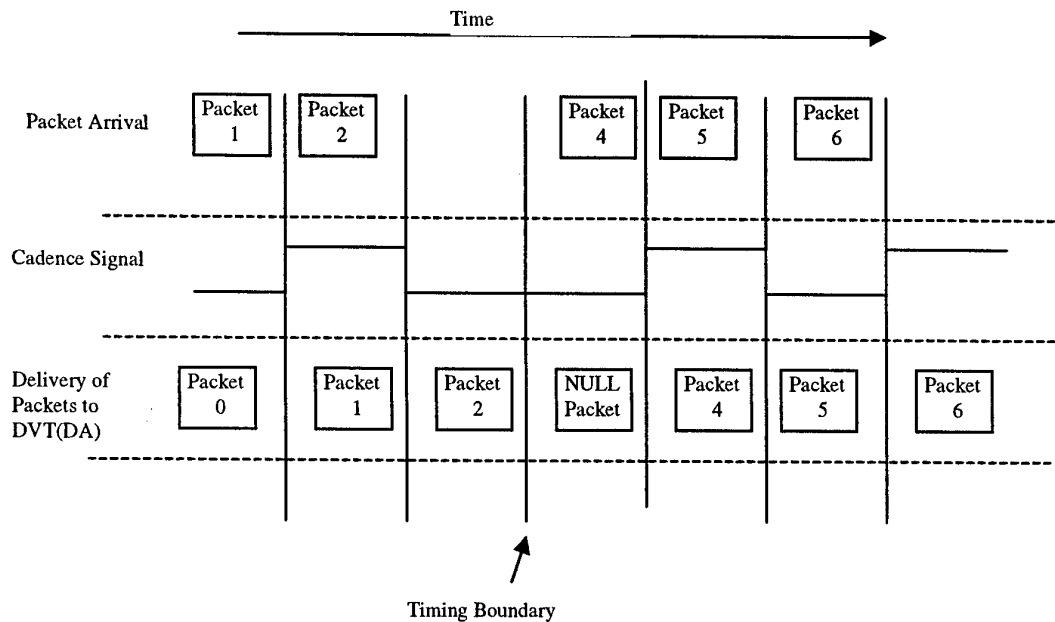


Figure 4 - The effect of packets being dropped by the network

If a packet is dropped by the ethernet then the PIC will eventually underflow the buffer and the read pointer will bounce back. Within the time resolution of the uCsim the cadence is still operating normally, ie at each packet arrival the cadence is inverting. However, the uCsim can see the packet sequence broken but it knows it can immediately send the current packet surmising that the PIC will be sending a null packet to fill the sequence gap.

2.3 Delayed Packets

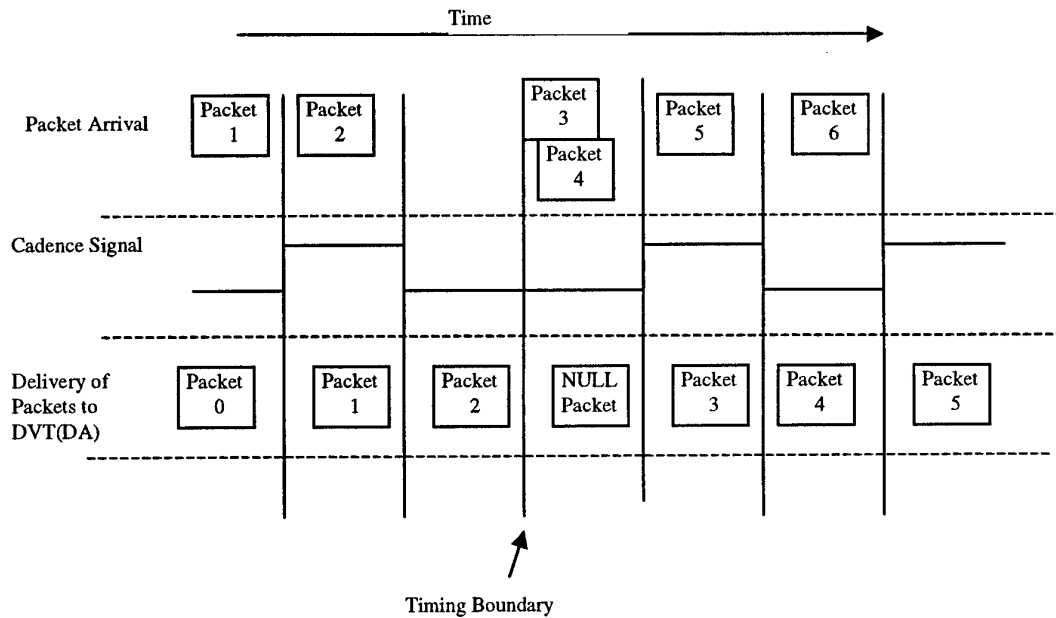


Figure 5 - The effect of late packets

This diagram shows the effect of a late packet. In the example packet 3 is delayed by a whole time increment (one packet worth of audio data, which is 35 msec). The uCsim does not have the time resolution to see that the packet is late, but the cadence signal indicates it can be written down to the PIC. When packet 4 arrives, the cadence signal has not changed so it is buffered in the uCsim, effectively adding an additional 35 msec of dejitter buffer. In due course the cadence resumes its rhythm and the process continues with an additional dejitter delay from the packet buffered in the uCsim.

2.4 Extremely Late Packets

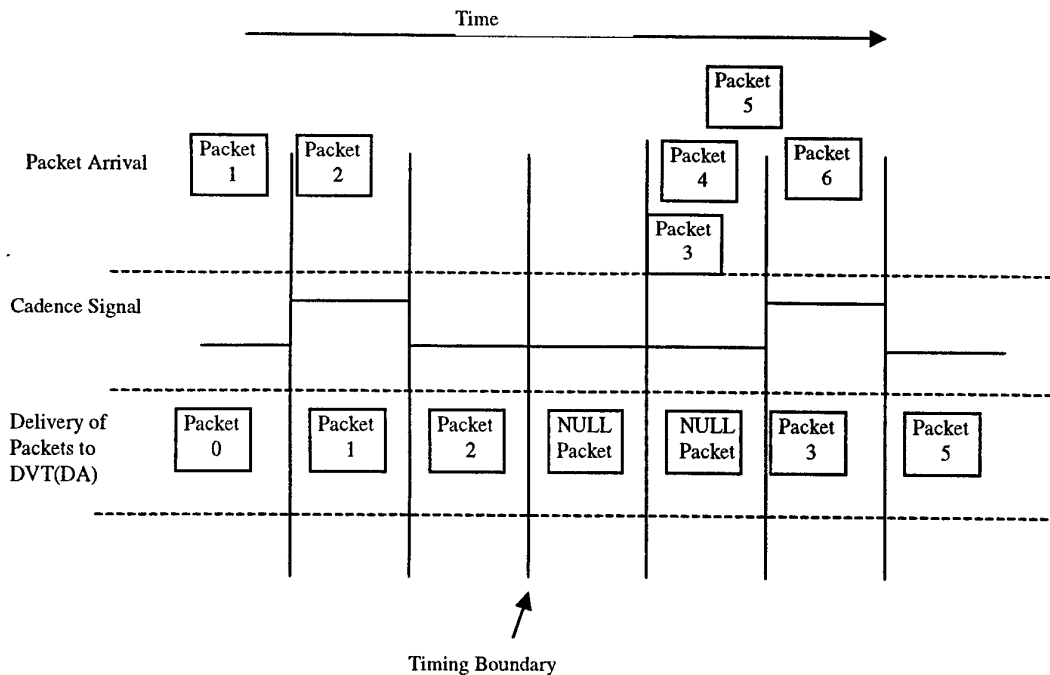


Figure 6 - The effect of extremely late packets on the device

This diagram shows the effect of multiple packets arriving extremely late. Ethernet is temporally constant so packets will arrive in the order that they are sent (in all but the most contrived situations). The order of the packets will always be increasing in sequence number (though packet loss may cause uneven increments). In the above situation, packets 3 and 4 have been delayed until the fifth packet's "timeslot" and then they all burst in. As soon as packet 3 arrives it is written down the line and then packet 4 is buffered awaiting the cadence signal change in level. To ensure that the end to end delay does not become excessive, the uCsim buffer is limited to one packet. When packet 5 arrives it replaces packet 4 in the buffer. This means that the packet 4 data has been dropped from the system and replaced by a null packet sent earlier by the PIC.

This effect may occur when the ethernet network is under a very high utilisation (i.e. when a file transfer occurs).

Appendix C: More Detailed Information

1. Detailed Design

1.1 Mitel Chip interface

The Mitel chip interface is derived from the standard design shown in the Mitel MT9172 specification. The only issue in adapting the design was locating a source for the special transformers required to unbalance the signal. The standard circuit design (as per the specifications sheet for the product) calls for a "2 to (1/2+1/2)" balanced transformer. No source for this component was found so two "2 to (1+1)" transformers (RS-Components 210-6346 pulse transformers) were used instead. The Mitel chip then demodulates the signal back into a 4 wire baseband TTL signal which is fed into the synchronous TTL to RS232 circuit.

1.2 16kb/s sync TTL stream to RS232

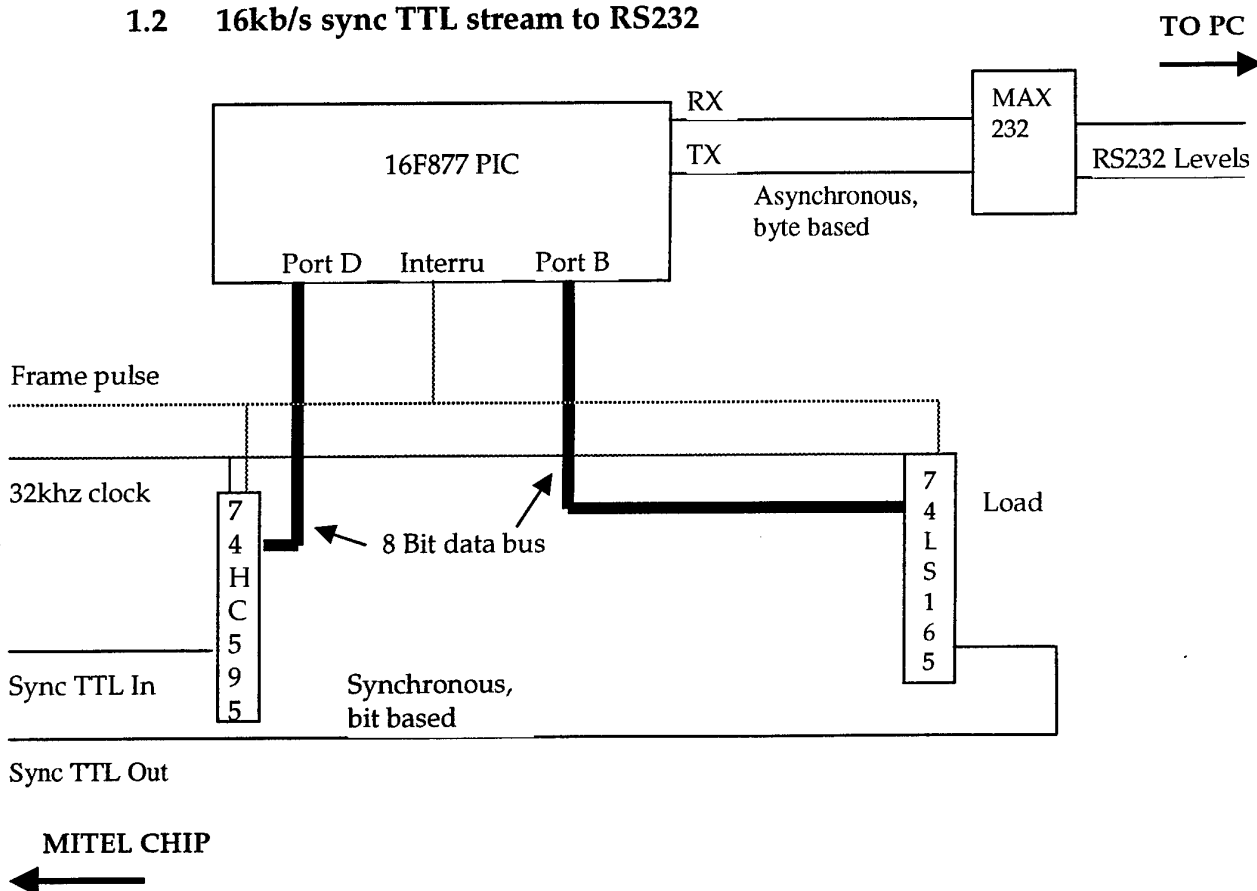


Figure 7 – Block diagram of the Synchronous TTL to RS232 sub-system

1.2.1 TTL Input to TX on RS232

The 16kb/s TTL level stream is clocked into a shift register with latches (in this case, a 74HC595) by the 32kHz clock produced by the Mitel chip. When the frame pulse is triggered (see below for a description of the frame pulse) the 74HC595 loads the contents of its internal shift registers onto its 8 output pins. The frame pulse also interrupts the PIC and the PIC then reads the pins of port D. This data is then put into an internal buffer and is sent out of the RS232 port of the PIC chip in an asynchronous manner. The MAX232 then level shifts the signal so any standard RS232 serial port can read it.

1.2.2 RX on RS232 to TTL Output

A character is transmitted by the uCsim down the RS232 line into the MAX232 chip. It is then level shifted to the TTL levels used by the PIC chip. The PIC chip then reads the RS232 character and copies it to an internal circular buffer. When an interrupt is received from port D (through the interrupt pin) the current character in the circular buffer is presented onto port B. These 8 bits are read by a 8 bit parallel to serial convert (for this circuit a 74LS165 is used) and copied to internal shift registers. These registered are then clocked out of the chip by the 32khz clock and onto the TTL out line.

1.2.3 Frame Pulse

The frame pulse is a signal that indicates the byte boundaries in the TTL stream. It is pulled low half way through each 8th bit. The PIC is a byte based machine, so this signal is used to convert the synchronous incoming bit stream into a byte orientated one.

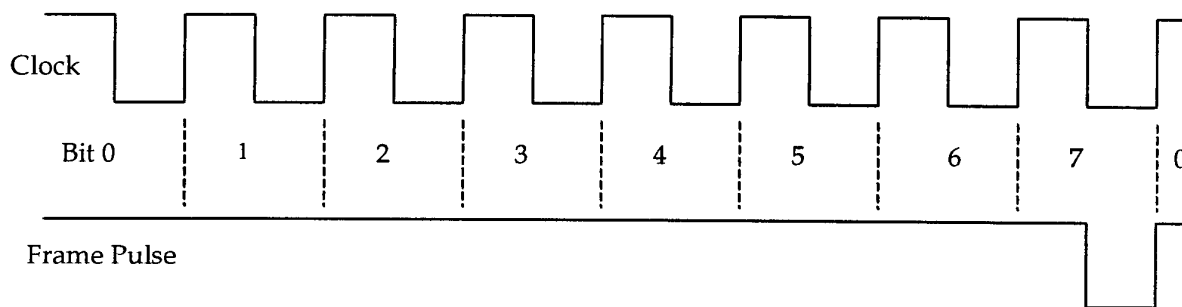


Figure 8 – The Frame Pulse Signal

This frame pulse is generated by the Mitel chip for this device, but it can be created by discrete logic if needed.

1.3 PIC Chip program

1.3.1 Operation

The aim of the PIC chip is to provide an interface between the serial port on a RS232 capable device and the 16kb/s synchronous stream that the Mitel chip requires. The PIC has two internal buffers, one for incoming and one for outgoing data. The incoming buffer (i.e. data from the Mitel chip) is 90 bytes long. The outgoing buffer (i.e. data to the Mitel chip) consists of two separate 70 byte buffers (see Cadence for details).

B.1.3.1.1 Buffering

The PIC has a unique outgoing buffering design. Writes into the buffer (from the RS232 port) are essentially packet based. Data written into the buffer will be in 70 byte bursts. Reads from this buffer are byte based, with one byte being read at a rate of 2khz (1/8 of 16khz).

This packet based transmission brought about the idea of the dual buffers and the idea of the PIC chip transmitting a heartbeat back to the uCsim. The dual buffers provide a buffer to read out of (to the Mitel chip) and a separate buffer to write into (from the uCsim/PC). If the read out buffer is emptied while the write in buffer is still being filled then the read in buffer loops back to the start thereby giving us a packet's worth of jitter buffer to use. A pin on the PIC chip is used to indicate which buffer the chip is currently reading from. This information is used by the uCsim to determine when to write down a new packet and when to buffer or drop a packet.

The program created for the PIC has two main procedures that will now be discussed.

B.1.3.1.2 Interrupt Service Routine

This procedure services port D (the parallel port of the PIC) and the receive line for the RS232 data. If a parallel port interrupt is received then it copies the data from port D the current input buffer (to uCsim/PC) and increments the pointer. It then checks where the current read out pointer is. If it is at the boundary of its current buffer (see Cadence for a diagram of the boundaries) the following algorithm is followed:

```

if (write out pointer is at start of next buffer)
    reset read pointer to start of current buffer
else
    switch read pointer to new buffer
  
```

Pseudo Code 1 - PIC algorithm for controlling the read pointers bounce back effect

If the interrupt is a RS232 interrupt then it copies the incoming character into the output buffer (to the Mitel circuit).

B.1.3.1.3 *Main Loop*

The main loop just scans the input circular buffer and sends any new data from it out of the RS232 TX port. This means that the buffer will be emptied whenever the PIC is not servicing the interrupt routine. The PIC has enough processing power that this buffer should never have more than 10 bytes waiting to be transmitted. The 56kb/s data rate to the RS232 port means that this buffer can be emptied 3 times faster than it can be filled so a backlog of data can not build up.

1.3.2 Extra Features

The PIC has an internal watchdog timer that detects program errors (crashes) and automatically resets the chip on such conditions. This makes the PIC robust to any possible error conditions, as it will just reset itself to a default state.

1.4 uCsim Design

The uCsim is a Linux based PC on a ram stick. It has a very small footprint and a large set of features. For this application, the important parts are the RS232 serial port and the 10baseT Ethernet connection. The uCsim receives bytes from the PIC via its RS232 port and then packetises them and sends them out of the Ethernet port. It also does the reverse to create full duplex communications. The reason the uCsim was chosen as the Ethernet interface device is twofold. The first is its flexibility. The uCsim runs the Linux OS so programs can be written quickly and efficiently. It also provides portability for the program if another Unix based platform is chosen to replace the uCsim. The second reason the uCsim was chosen is the small footprint of the device. It should be possible to eventually package the whole PVC unit into a box about the size of a cigarette packet.

The interface program and ancillary programs will now be discussed.

1.4.1 The uCsim

The main logic of the program is best described by this pseudo code:

```
while (forever) {
    wait for ethernet port or serial port to have data;
    if( ethernet port ready) {
        read packet from port;
        if ( value from PIC heartbeat different from
last time) {
            if ( no currently buffered packet) {
```

```

        write packet out of serial port;
    } else {
        write buffered packet out of serial
port;
        copy current packet to buffer;
    }
    } else {
        copy packet to buffer;
    }
} // end of if(ethernet port ready)

if ( buffered packet AND value from PIC heartbeat different
from last time) {
    write out buffered packet to serial port;
}

if ( serial port ready) {
    read bytes available from serial port to incoming
buffer;
}

if ( packet worth of bytes in serial buffer) {
    if ( buffer too full) {
        drop packet worth of bytes;
    }
    create ethernet packet with buffer;
    send packet;
}

} // end of while(forever)

```

Pseudo Code 2 - The main loop for the uCsim logic

This program uses the heartbeat from the PIC to throttle the delivery of packets to the PIC. This method was chosen because of the inaccurate timing available in the uCsim and because of the very low processing power needed to implement this design.

The loop has two main sections, the section for handling the arrival of packets from the Ethernet and the code dealing with reading of bytes from the serial port. The main loop starts with a select() function call. This function waits for a file descriptor to become ready to be read from. When one or more is available the call returns and execution continues.

The first section handles the arrival of Ethernet packets. When a packet arrives, it is read into a memory buffer and a check is performed to determine whether it should be written out of the serial port. If the state of the heartbeat pin on the PIC has changed

from last time a packet was written, then packet is written to the serial port if no other packet is currently buffered. If one is buffered then the buffered packet is written to the serial port and the new packet is put into the buffer in its place. However, if the heartbeat pin hasn't changed since the last packet the new packet is buffered. The packet buffer is only one packet deep, so when a new packet is buffered any packet currently in the buffer are effectively discarded. There is one final check. If the heartbeat pin has changed and if there is a buffered packet, that packet is written to the serial port. This check is performed because it is possible for packets to be delayed for long periods, so the writing of any buffered data still needs to be performed if a new packet hasn't arrived.

The second section of the code deals with packetising the data arriving from the DVT(DA). The first if() statement copies any available data from the serial port into a linear buffer. Next, a statement checks if there is one packet worth of data currently in the buffer. If there is enough data a packet is constructed and sent out of the Ethernet port. The number of bytes of data in the incoming buffer is also checked. If this buffer gets larger than two packets worth of bytes data is dropped. If this didn't happen it would be possible for the two ends of the connection to slowly slip out of time with each other as more and more data is buffered.

1.4.2 Kernel

There was one issue with the Linux kernel running on this device. The serial port code had a section that reacted to a certain byte value being sent down the line. The problem was that the DVT(DA) would produce this character under normal operating conditions. So, the source code of the kernel was altered to remove this functionality.

2. Differences Between Switch/Multiplexer End and DVT(DA) End

Two differences exist between the switch and DVT(DA) mitel devices. This difference is due to the way the balanced signal is processed by the transformers to produce the unbalanced signal. The balanced signal being sent from the switch has a 50V bias on one of the wires, whereas the DVT(DA) sends an unbiased signal. The signal from the switch/DVT(DA) is processed through two transformers to provide an unbalanced signal. A schematic is shown below:

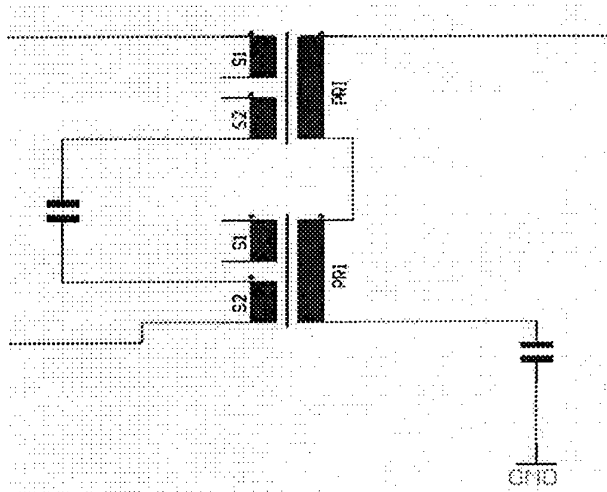


Figure 9 - Transformer interface

Due to the bias on the switch balance line it does not need an external voltage input to provide the voltage differential to the transformers. It does however require a non-polarised capacitor to be used so that the input can be polarity insensitive. The DVT(DA) end does not have this built in bias so it uses a polarised capacitor on the left and it also has the ability to add a voltage bias to the transformers via an external voltage source. This would only be needed where there are large distances between the parakeet digital phone and the PVC device.

3. Device Cost Breakdown

There are three major cost areas in this device, the uCsim, the PIC chip and the Mitel MT9172. This table summaries the costs for each area:

Section	Approximate (\$AU)	Cost
Mitel		
MT9172	30	
Transformers	10	
PIC		
PIC16F877	25	
TTL Logic	3	
MAX232	5	
uCsim		
uCsim	300	
Support board	40	
TOTAL	420	

Table 1 - PVC Costings

The most expensive part of this design is the uCsim, which is a complete embedded PC. For a production device, it could be possible to replace the uCsim functionality with customised hardware. This would reduce the cost per unit of the device, but the development cost for the design of the specialised hardware could outweigh the cost per unit savings. Another cost saving measure could be removing the PIC from the design and then using the uCsim to implement its functionality. This may be possible if the Real-Time Linux patches are used, but the development costs involved in implementing this functionality could easily be more than the cost savings.

Also, note that these costs are the price paid for the small purchases required for the prototype. A production version of this device would be able to benefit from the savings involved in bulk material purchases.

4. Issues for Inclusion in Operating Procedures

PVC adds additional end to end latency. This is generally not a problem except in cases where the DVT(DA) and circuit switch are involved in lengthy handshaking. For instance when the handset is lifted in preparation for dialing, the DVT(DA) and switch exchange around 20 messages each of which incurs a 35 msec delay. If the user attempts to enter a subscriber number before this handshaking is complete then the dialed digit will be lost. Accordingly, operating procedures must emphasise the need to obtain a dial tone before commencing to dial.

There is another issue relating to the order in which the devices are powered up. The Parakeet switch will only recognise a phone if it is powered up after the switch itself. This is not an issue with the currently fielded Parakeet system as all the phones are powered by the switch itself. However, as the PVC provides independent power for the phone it causes a problem. The PVC device at the switch/multiplexer end must be powered up before the PVC at the DVT(DA) end. There must also be at least 30 seconds delay between the switch end being powered up and the DVT(DA) end. This provides enough time for the switch end to initialise itself and to start sending a valid signal to the phone end of the "tunnel". If this timing issue is ignored the DVT(DA) will fail to produce a dial tone and the unit will not work. Turning the power off and then on again (power cycling) at the DVT(DA) end will solve this problem.

There is a final issue relating to buffering and the device. There is an orange indicator light on the PVC box that gives an indication of the state of the device. When it is constant (always on or off) it signifies that the PVC has locked into a valid stream of ethernet data. If the LED is flickering it signifies that the device is unable to find or lock into a valid stream. If this flickering occurs during operation then pressing of the reset switch on the front of the device should force the PVC to resynchronise into the data stream. This resetting of the device can be done multiple times to attempt to get a good connection. This reset can be performed at either end of the link to try to regain synchronisation for each end, but it is usually more successful if attempted on the end with the flickering light. There may be artifacts heard on the PVC connection during

the first 3 minutes of the devices operation. This is due to the Mitel chipset training onto the signal sent by the DVT(DA).

5. Future Design Work

1. Switch Server negotiation

The current PVC devices are each hard-coded into pairs. Each PVC has hard-coded into it the ethernet media access control (MAC) address of the other end of the PVC link. An alternative approach would be to allow dynamic learning of the MAC addresses.

An approach that could be explored would be by using a master-slave relationship (this relationship is totally separate to the master-slave relationship of the Mitel devices). Devices attached to the multiplexers will be designated as masters. These devices listen to the ethernet network and respond to client requests to construct a data stream.

Each client will have a (mux,port) pair assigned to it (via the configuration interface). When a client starts up it sends a broadcast ethernet packet asking for the server endpoint for its (mux,port) pair to respond. The server end receives this broadcast and then responds with an acknowledgement packet and a data stream is constructed.

The (mux,port) assignment of PVC devices will be controlled by the switch management team, with a set of devices assigned to a particular telephone number. By using this (mux,port) mapping, a telephone number will be able to follow a user around. The user will just plug their device into a new ethernet location and calls will be routed to them.

2. Device configuration

Configuring the PVC is currently carried out via a serial link and telnet terminal session. An alternative method which could be developed would be a web interface using the common gateway interface (CGI) on a web server hosted on the uCsim on the uCsim. By using this (mux,port) mapping, a telephone number will be able to follow a user around. The user will just plug their device into a new ethernet location and calls will be routed to them.

Configuration changes should rarely need to be done if the server discovery was automatic. The only changes that may be needed are changing the device from a client to a server (depending on what type of device it is attached to).

3. Cabling

The current PVC expects to be connected to its own ethernet connection. For the concept demonstrator a small hub is used so that the BCSS workstation can share a

single connection to the BCSS LAN with the PVC. A possible development would incorporate a hub inside the PVC casing and the users PC would then daisy chain off that. That is for the DVT(DA) end, where the DVT(DA) sits on the commander's desk, and the PVC box sits atop his computer.

It is unlikely that a BCSS workstation would be located with the Parakeet switch so a dedicated ethernet switch would be required for the devices.

The ethernet interfaces can be visualised by the following diagram:

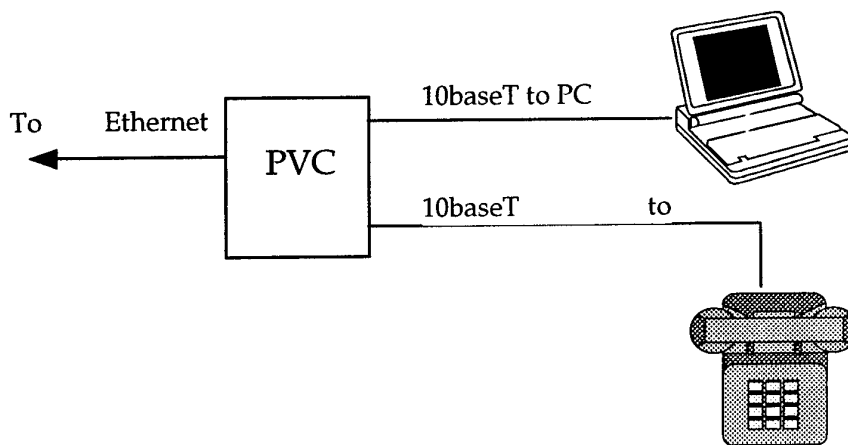


Figure 10 PVC Cabling Configuration

This reduces the clients available bandwidth to the network by whatever the PVC uses. Since the DVT(DA) will only generate 16kb/s of traffic (plus Ethernet packet headers overhead) this should not be a problem as the BCSS environment provides 10mb/s of switch ethernet to the desktop.

4. Loop Group PVC

An extension to this 1 to 1 mapping of DVT(DA)'s to terminating devices is the construction of a loop group device. This box will physically replace a Parakeet loop group multiplexer and connect it directly to the ethernet. The device would access the 32 time slots of a Parakeet loop group bridging all 30 traffic channels over the ethernet. At the DVT(DA) end a normal PVC device would be employed.

This could substantial reduce the cost of the PVC implementation as the loop group device is unlikely to cost 30 times the cost of the individual PVC device.

A key technical issue is that this will require the server to be able to handle a 512kb/s stream on the synchronous output. The buffer size for this device will have to be carefully considered so the trade off between latency and bandwidth usage can be made. Also, while the TDM structure of a loop group is well known, there are no standard definitions for the use of the signalling time slot. This slot is reserved in the Eurocom standard for "national use" and the manufacturers of the Parakeet circuit switch may be using this channel in an unknown fashion.

DISTRIBUTION LIST

Parakeet Virtual Cable Concept Demonstrator

A.B. Reynolds and W.D. Blair

AUSTRALIA

DEFENCE ORGANISATION

Task Sponsor DGC4

S&T Program

Chief Defence Scientist	} shared copy
FAS Science Policy	
AS Science Corporate Management	
Director General Science Policy Development	
Counsellor Defence Science, London (Doc Data Sheet)	
Counsellor Defence Science, Washington (Doc Data Sheet)	
Scientific Adviser to MRDC Thailand (Doc Data Sheet)	
Director General Scientific Advisers and Trials/Scientific Adviser Policy and Command (shared copy)	
Navy Scientific Adviser (Doc Data Sheet and distribution list only)	
Scientific Adviser - Army	
Air Force Scientific Adviser	
Director Trials	

Aeronautical and Maritime Research Laboratory

Director

Electronics and Surveillance Research Laboratory

Director (Doc Data Sheet)

Chief of Communications Division
Research Leader, MIN Branch
Head Network Architecture Group
Head Network Management Group
Head Wireless Systems Group
W.D. Blair
A.B. Reynolds
J.T. Ball

Chief of Land Operations Division
Research Leader, LS Branch
F. Bowden

DSTO Library

Library Fishermans Bend
Library Maribyrnong

Library Salisbury
Australian Archives
Library, MOD, Pyrmont (Doc Data sheet only)
US Defence Technical Information Center, 2 copies
UK Defence Research Information Centre, 2 copies
Canada Defence Scientific Information Service, 1 copy
NZ Defence Information Centre, 1 copy
National Library of Australia, 1 copy

Capability Systems Staff

Director General Maritime Development (Doc Data Sheet only)
Director General Land Development
Director General Aerospace Development (Doc Data Sheet only)

Knowledge Staff

Director General Command, Control, Communications and Computers (DGC4)
(Doc Data Sheet only)
Director General Intelligence, Surveillance, Reconnaissance, and Electronic
Warfare (DGISREW) R1-3-A142 CANBERRA ACT 2600 (Doc Data Sheet
only)
Director General Defence Knowledge Improvement Team (DGDKNIT)
R1-5-A165, CANBERRA ACT 2600 (Doc Data Sheet only)

Army

Stuart Schnaars, ABCA Standardisation Officer, Tobruk Barracks, Puckapunyal,
3662 (4 copies)
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), MILPO Gallipoli
Barracks, Enoggera QLD 4052 (Doc Data Sheet only)
NPOC QWG Engineer NBCD Combat Development Wing, Tobruk Barracks,
Puckapunyal, 3662
Army Communications Training Centre

Intelligence Program

DGSTA Defence Intelligence Organisation
Manager, Information Centre, Defence Intelligence Organisation

Acquisitions Program

PD.Parakeet
PD BCSS
PD DJFHQ (Afloat)
Land Engineering Agency (2 Copies: Library and G. Lampard)

Corporate Support Program (libraries)

Library Manager, DLS-Canberra

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy
Library
Head of Aerospace and Mechanical Engineering

Deakin University, Serials Section (M list), Deakin University Library, Geelong,
3217 (Senior Librarian, Hargrave Library, Monash University (Doc Data
Sheet)
Librarian, Flinders University

OTHER ORGANISATIONS

NASA (Canberra)
AusInfo
State Library of South Australia
Parliamentary Library, South Australia

OUTSIDE AUSTRALIA

ABSTRACTING AND INFORMATION ORGANISATIONS

INSPEC: Acquisitions Section Institution of Electrical Engineers
Library, Chemical Abstracts Reference Service
Engineering Societies Library, US
Materials Information, Cambridge Scientific Abstracts, US
Documents Librarian, The Center for Research Libraries, US

INFORMATION EXCHANGE AGREEMENT PARTNERS

Acquisitions Unit, Science Reference and Information Service, UK
Library - Exchange Desk, National Institute of Standards and Technology, US

SPARES (5 copies)

Total number of copies: 65

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA					
				1. PRIVACY MARKING/CAVEAT (OF DOCUMENT) nil	
2. TITLE Parakeet Virtual Cable Concept Demonstrator			3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION) Document (U) Title (U) Abstract (U)		
4. AUTHOR(S) A.B. Reynolds and W.D. Blair			5. CORPORATE AUTHOR Electronics and Surveillance Research Laboratory PO Box 1500 Salisbury SA 5108 Australia		
6a. DSTO NUMBER DSTO-TR-1113		6b. AR NUMBER AR-011-778		6c. TYPE OF REPORT Technical Report	
				7. DOCUMENT DATE March 2001	
8. FILE NUMBER E8730/15/15		9. TASK NUMBER 99/141		10. TASK SPONSOR DGC4	
				11. NO. OF PAGES 32	
				12. NO. OF REFERENCES 2	
13. URL ON WORLDWIDE WEB http://www.dsto.defence.gov.au/corporate/reports/DSTO-TR-1113.pdf			14. RELEASE AUTHORITY Chief, Communications Division		
15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT <i>Approved for public release</i>					
OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE CENTRE, DIS NETWORK OFFICE, DEPT OF DEFENCE, CAMPBELL PARK OFFICES, CANBERRA ACT 2600					
16. DELIBERATE ANNOUNCEMENT No Limitations					
17. CASUAL ANNOUNCEMENT Yes					
18. DEFTTEST DESCRIPTORS Parakeet project. Mobile communication systems.					
19. ABSTRACT Project Parakeet is deploying a secure military telephony system into Australia's land forces. The Battlespace Command Support System (BCSS) project is deploying local area networks (LAN) and computers to these same forces. This report describes a concept demonstration that permits the use of the BCSS LAN for the connection of Parakeet telephones to the Parakeet circuit switches rather than using separate cabling. Such a device would provide for faster setup/teardown of deployed headquarters and reduce the demands on linesmen to lay cable within the headquarters.					